Journal of Intelligent & Fuzzy Systems 39 (2020) 6009–6020
DOI:10.3233/JIFS-189075
IOS Press

A flexible authentication scheme for smart home networks using app interactions and machine learning

Yosef Ashibani* and Qusay H. Mahmoud
Department of Electrical, Computer and Software Engineering, Ontario Tech University, Oshawa, Ontario, Canada

Abstract. Smartphones have now become ubiquitous for accessing and controlling home appliances in smart homes, a popular application of the Internet of Things. User authentication on smartphones is mostly achieved at initial access. However, without applying a continuous authentication process, the network will be susceptible to unauthorized users. This issue emphasizes the importance of offering a continuous authentication scheme to identify the current user of the device. This can be achieved by extracting information during smartphone usage, including application access patterns. In this paper, we present a flexible machine learning user authentication scheme for smart home networks based on smartphone usage. Considering that users may run their smartphone applications differently during different day time intervals as well as different days of the week, new features are extracted by considering this information. The scheme is evaluated on a real-world dataset for continuous user authentication. The results show that the presented scheme authenticates users with high accuracy.

Keywords: App interactions, user authentication, smart home network, classification, foreground apps

1. Introduction

The smart home, a popular application of the Internet of Things (IoT), refers to a home equipped with an automated system that includes temperature monitoring, alarms and cameras. In addition to accessing, operating, and controlling home appliances, smart home networks, as shown in Fig. 1, provide many other services to home residents, such as entertainment storage information, personal files, and family photos. As an example, Samsung's SmartThings and HomeKit are smart home platforms where control management and authentication are mostly performed through an installed application on end-user devices.

Although such home automation systems provide convenience to home residents, they provide security challenges that need to be considered. One of these challenges is user authentication. Most access is achieved through the Internet while homeowners are outside their homes. Furthermore, access to such services is mostly achieved remotely through the users' smartphones, which have become an integral part of most smart home systems. Thus, smart homes should be enhanced with security measures that ensure that the access request and control commands come from the authorized user. Many smartphones and tablets are currently prepared with authentication methods, such as fingerprint readers, for accessing devices or their applications (apps). However, some users might avoid setting short access ses-

*Corresponding author. Yosef Ashibani, Department of Electrical, Computer and Software Engineering, Ontario Tech University, Oshawa, Ontario, L1G 0C5 Canada. E-mail: yosef.ashibani@ontariotechu.net.

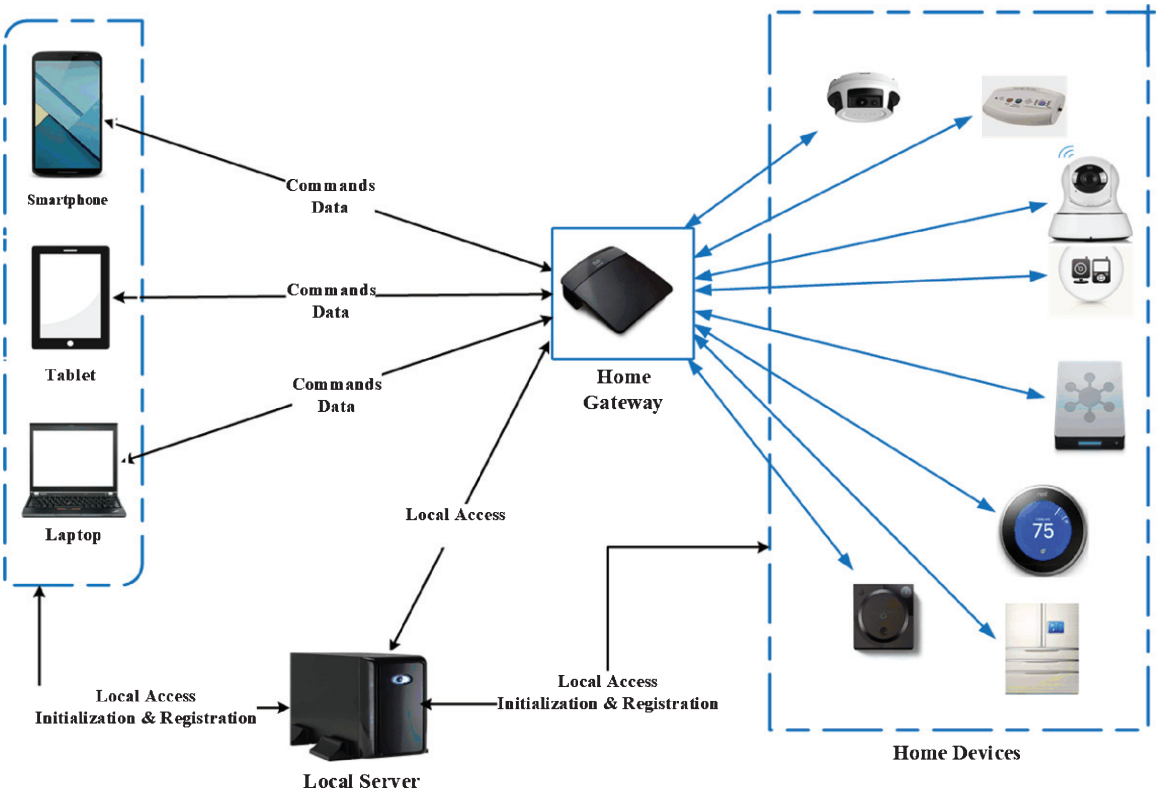

Fig. 1. Smart home network.

sions after initial authentication. As a result, after the entry point, devices are vulnerable to use by others. Therefore, it is crucial to present an authentication scheme that overcomes these issues and can be implicitly and continuously utilized without the need for explicit user intervention. Consequently, any security solution should include user authentication at the access request time; continuous user authentication beyond the log in stage on the mobile device; and the ability to both block access requests that come from unauthorized users and inform the homeowner. Many proposed solutions do not consider access sessions based on previous accessed events.

Hence, a continuous authentication scheme based on mobile app usage behavior is required as a second layer of authentication for smart homes. This research is motivated by the following factors:

- The number of apps in use has dramatically increased, while smartphones/tablets have become an essential tool in accessing smart home networks.
- Many apps run in the foreground mode, which reflects user interaction with them, hence enabling the user's usage behavior to be modeled for user authentication.
- In most cases, access to app sessions on end-user devices is short, which means that users continuously switch between apps within a short period [1], allowing access behavior to be drawn as a result. Additionally, diversity of app access sessions can be applied for continuous user authentication.
- The improvement in the running functionality of smart home controllers (hubs), such as Samsung Connect Home hub and HomeKit hub [2], Wink hub and Insteon hub [3], means that they can handle an advanced continuous authentication mechanism in the background, based on user access patterns on end-devices.

This paper, which is an extension of a previous work in [4], presents an authentication scheme based on machine learning, utilizing user behavior while accessing apps on smartphones. To this end, the main

contribution of this paper is a machine learning-based continuous user authentication method for smart homes, utilizing app access interactions. This method:

- Authenticates registered users utilizing app interactions on smartphones with a considerable low false positive rate (FPR) and false negative rate (FNR);
- Uses implicit features that can be collected (or extracted) in the background without requiring user intervention during the extraction or in the authentication process;
- Does not require new training when having a new end-device or where the previous device is stolen or lost; hence, presenting flexibility to changes in the user usage as the proposed solution is centralized, and the previously built user template pattern is already available at the home hub;
- Does not require retraining the model when a new app is added by the user, and subsequent model training will be achieved only on the newly added apps.

The rest of this paper is organized as follows. Section 2 discusses the related work. The proposed scheme is presented in Section 3, while Section 4 presents the evaluation and results. Section 6 concludes the paper and offers ideas for future research.

2. Related work

Many authentication approaches that apply device resources and mobile apps have been presented in the literature. For example, in applying built-in battery voltage sensors, the authors in [5] present a power model construction approach that monitors power consumption per app on user devices. However, it is challenging to model power consumption only for specific apps due to other apps running in the background. The authors in [6] show that utilizing text messages and calling behavior enables users to be identified while anomalies can be detected based on their interactions with their mobile apps. A user profile approach that gathers behavioral information, including calls, text messages and geographic location, is proposed in [7]. The study in [8] proposes an anomaly-based detection approach, using machine learning based on monitoring users' actions, such as sending text messages or calls.

A further study [9] presents an authentication model based on app usage behavior. The features employed in this study include app name, action name, and length of message as well as email, call duration and time. A study in [10] evaluates text messages, apps, accessed websites and location for user identification according to classification, based on approximately 30 days from the dataset. An associative classification-based user authentication approach is reported in [11]. In this research, collected network data, including Bluetooth and Wi-Fi, achieved an accuracy of 91% by combining apps and Wi-Fi data. The authors in [12] utilize two derived features, namely email access time and location, from mobile devices. The work in [13] presents a continuous authentication model for smartphones based on app usage data. Although the presented work does not consider utilizing apps by all users, it includes all apps as well as considers those that are used only by individual users. A behavior profiling technique, which applies historical app usage that continuously verifies mobile users, is proposed in [14]. However, the time frame is limited to a maximum of 22 days. In [15,16], user authentication approaches that are based on app access patterns, as well as generated traffic while accessing these apps, are introduced.

A user authentication method that uses a gesture recognition algorithm is proposed in [17]. This model utilizes user touch screen interactions in addition to device feedback vibrations. Continuous authentication for smartphones based on smartphone gesture patterns is presented in [18]. The authors in [19] determine the possibility of developing new user authentication mechanisms by utilizing multi-touch gestures. A study of utilizing and analyzing gait data is made in [20]. This study involved 11 volunteers, with an achieved accuracy of 79.1% and 92.7%, respectively. The gait characteristics of a user, based on six gait signature metrics according to the rate of changes of acceleration data, is presented in [21].

In [22] and [23], the authors described a method, also based on gait recognition, for determining whether the owner is using the device. Authenticating users of a smartphone according to accelerometer-based gait recognition, using the k-NN algorithm, is proposed in [24]. This approach, which records data as the user is walking, is built on the assumption that different individuals have different walking patterns. This method needs 30 seconds for authentication and requires users to follow a script.

As can be seen from the literature, many studies do not consider extracting more features in addition to

not considering shared apps or separately considering weekdays and weekend access time. In addition, the evaluation was conducted on only a small number of users. Moreover, the focus of the earliest studies is on behavior misuse discovery during mobile network interactions, including calling and messaging services such as in [25,26]. However, as the number of mobile apps increases, traditional text messages and calls are being replaced with apps that achieve the same purpose.

This idea of behavior user authentication has been utilized within different technologies, such as mobile networks and web browsing, for the client or server side [27]. For example, the studies in [28–30] utilize features while accessing a computer system, such as visited websites and accessed files to detect unauthorized access to the system. The literature presents approaches that use a specific app to discriminate between users, but our goal is to utilize features to continuously authenticate users.

3. Proposed authentication scheme

This section presents a flexible machine learning-based authentication scheme, which first learns users' app access behavior on their smartphone devices and then continuously authenticates them, based on the learned pattern template. This scheme is considered a flexible for the following reasons: (1) uses implicit features that can be extracted in the background without requiring user intervention during the extraction; (2) does not require new training when having a new smartphone; hence, presenting flexibility to changes in the user usage as the proposed solution is centralized; (3) does not require complete retraining the model when a new app is added by the user, and subsequent model training will be achieved only on the newly added apps. The architecture of the proposed scheme is presented in Fig. 2. The goals that are considered in the designed solution are to:

- Implicitly authenticate users with a high level of accuracy;
- Offload the computation of user pattern template building of the mobile end-user device to the smart home controller;
- Utilize implicit features that can be collected (or extracted) in the background with minimal power consumption, without requiring user intervention in the identification process.

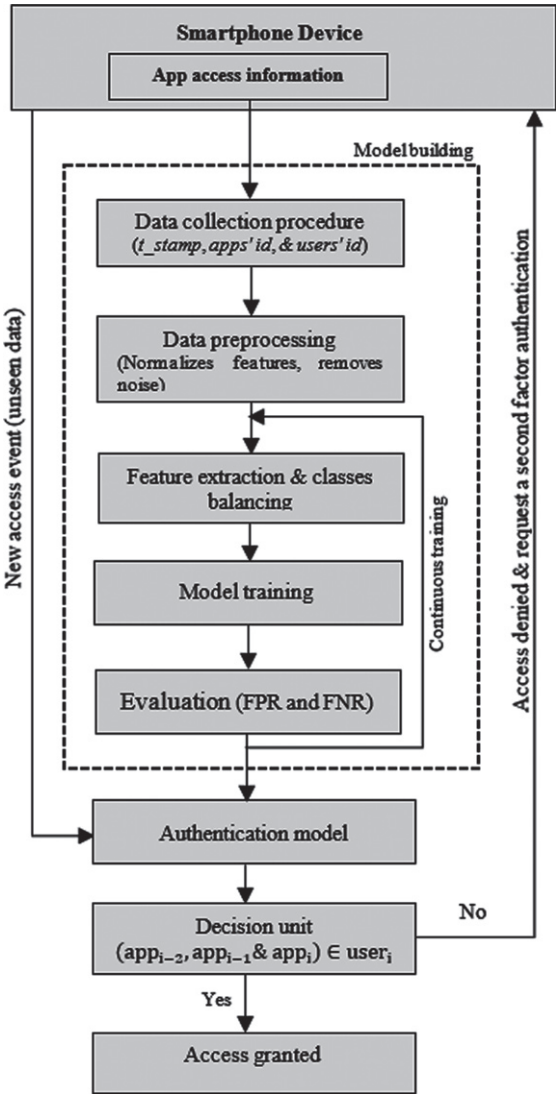

Fig. 2. Proposed authentication scheme.

3.1. Threat model

Accessing home networks and controlling home IoT devices is mainly achieved via smartphones or tablets of pre-registered users. However, access can be achieved by other users, such as visitors, friends, and relatives through known user's smartphones. Consequently, there is a main security point where unauthorized insider access to the home network could occur. This access can be achieved through a registered authorized user's device by another authorized user (insider), with the latter's unauthorized access permission to a part of the home services. Such a user might intentionally or unintentionally achieve

the user's smartphone login credentials, hence be able to log in to the smartphone and be able to use apps on that smartphone, including the smart home network app.

3.2. Scheme description and assumptions

A smart home system framework has been previously implemented in [31]. In this framework, users interact with smart home hubs that are assumed to be trusted, using their smartphones to access and control home appliances. The authentication scheme presented in this paper can be integrated into the framework, and this method works by tracking app access patterns on smartphone devices in order to detect any abnormal patterns while accessing smart home networks. In the following subsections, we describe the stages of the method presented in this paper.

3.2.1. Data collection, preprocessing and feature extraction

The app access history information is collected by smart home hubs for training, testing and then authentication phases. Thus, after being collected on the user's smartphone and temporarily stored in a first-in-first-out memory buffer, the presented method receives app access logs from the user's smartphone devices and extracts the suitable features. The next step is to build an authentication scheme; therefore, we need to consider the most efficient features that distinguish between users over time. Hence, we need to extract features from app events based on event-driven actions, meaning that data will only be collected when the user interacts with apps and features will then be extracted. A user's routine in accessing apps is usually in time intervals, but this sometimes deviates due to different circumstances. For instance, a user browses an app at the same time every day. However, due to a change in schedule (plan or routine), the app was checked late. Therefore, app access time is not always consistent. The interaction time in this study is considered access time to apps in the foreground, which reflects the user interaction with the smartphone devices during the access sessions. As a result, precise app access time might not always be consistent. Accordingly, the focus should be on access duration rather than on the initial access time. Hence, we divided the time of the day into six-time intervals and divided the day of the week into three weekday intervals (beginning of the week, end of the week, and weekend):

- time_interval_1 (>=00 : 00 am & <07 : 00 am);
- time_interval_2 (>=07 : 00 am & <10 : 00 am);
- time_interval_3 (>=10 : 00 am & <12 : 00 pm);
- time_interval_4 (>=12 : 00 pm & <17 : 00 pm);
- time_interval_5 (>=17 : 00 pm & <21 : 00 pm);
- time_interval_6 (>=21 : 00 pm & <00 : 00 am);
- week_interval_1;
- week_interval_2;
- week_interval_3.

Hence, the received access log which includes the timestamp (t_stamp) at the app access start time, the app ID and the user ID will be transferred to an event information that includes the generated new features. Hence, the timestamp is mapped onto one of the six-time intervals and one of the three weekday intervals. As well as considering the previously accessed app, the features utilized in this work include user ID; app ID; session start access time; session end access time and duration; and day of the week:

t_stamp_current ->(time_interval_i, week_interval_i)

3.2.2. Classifier training and imbalance handling

The dataset is divided into training and testing, while the oversampling [32] is applied only to the training dataset. An appropriate classifier will then be applied to events, with the prepared features from the previous step. For building the complete behavior pattern template, a binary classification strategy is adopted for providing authentication. As we are targeting multi-user authentication, one-vs-rest classification will be applied for each class, with the result that each access event will be classified as related to one of the enrolled known users, or not, as the case may be. As a preliminary experiment, different classification strategies are applied, including algorithms such as K-Nearest Neighbor (*KNN*) [33], GaussianNB (*NB*) [34] *MLP* Classifier (*MLP*) [35], Nearest Centroid (*NC*) [36], Random Forest Classifier (*RF*) [37]; and Support-Vector Machine (*SVM*). Algorithm 1 shows the procedure of selecting the best classifier.

Because it presents high accuracy, we chose to utilize the *RF* classifier. Even though other classifiers, such as *SVM*, have been used in the literature, it requires more computation time and produces less accuracy. Furthermore, RF offers advantages over other algorithms, such as *KNN*, *NB* and *SVM*, including the capacity to: overcome the problem of

Algorithm 1: Classifier Selection

INPUT: Dataset
OUTPUT: best user's classification model Cl_M_i.

procedure ()
input Dataset ← D.
select only foreground app-based interactions;
read features ← $\{f_1, \ldots, f_n\}$: of access time (D^{time});
normalize features, remove noise;
for Classifier (Cl_i). ← 1 to N **do** ▷ where N is the number of classifiers and (Cl_I) is the classifier I
    split D into D_i^{train}, D_i^{test}
    apply cross-validation only on the training data;
    for data_sample ← 1 to **k do** ▷ where k is the number of samples
        up-sampling the D_i^{train} of the minority class ($D_{main}^{up\_samp}$) of D^{time};
        train on data of $\sum_{i=1}^{i-1} week\,(s)$ and test on $week_{i+1}$
        test the Cl_M_i on D_i^{test}; ▷ where Cl_M_i is the classification model using classifier i
        build the model (Cl_M_i) using both $D_{min}^{up\_samp}$ and majority Class D_{maj};
        calculate FPR and FNR; ▷ where FPR is false positive rate, FNR is the false negative rate
    end for
end for
compare $Cl\_M_1, \ldots.., Cl\_M_N$
return the best Cl_M_i
end procedure

overfitting; offer less sensitivity to outliers in training data; measure the importance of every feature during training [38]; and provide accurate predictions on many types of applications [39]. Furthermore, utilizing the *RF* classifier most often prevents overfitting.

For training, the presented method receives the app access logs and preprocesses them in addition to considering any imbalanced representation of classes among class observations. To prevent considering majority over minority classes, class variance should also be considered. Many real-life classification scenarios, such as intrusion detection in networks, fraud detection, and health care diseases, have imbalance in the provided data. Imbalance in data means that classes are not with the same values. For example, one may include minority samples while the other(s) may include the majority samples of the data. In addition, the type of data in the application should be taken into account when imbalance is present in the data. As the one-vs-rest classification methodology is selected, the over-sampling method is applied to balance the class distribution of the data samples during the data training.

$$dec_i = \begin{cases} if\ (app_{i-2}, app_{i-1} and\ app_i) \in user_i, & grant\ access \\ if\ (app_{i-2}, app_{i-1} \in user_i)\ and\ (app_i \notin user_i), & deny\ access \\ if\ (app_{i-2}, app_{i-1} \notin user_i)\ and\ (app_i \in \mathrm{user_i}), & deny\ access \end{cases} \quad (1)$$

3.2.3. User authentication and decision unit

The authentication process requires running only one classification model (Cl_M_i), as shown in Fig. 3, on the information received from the registered device from which the request is issued. Therefore, each classification model enables the authentication of the related (assigned) user ($user_i$).

Hence, each user's information template is created by training the classifier on the given information of this user as a legitimate user while considering the rest of the users as illegitimate. Each classifier is trained on one user's data; thus, we need to construct N binary classifiers based on the number of users based on equation (2):

$$Auth\_M_i = (Cl\_M_1, Cl\_M_2, Cl\_M_3, \ldots\ldots Cl_{M_N}) \quad (2)$$

During the training phase, the classifier identifies each access event received from the end-user's device. However, this step may increase False Positive Rates (FPRs). Thus, to eliminate this issue, a number of specific events in determining access decisions should be considered. Consequently, we consider applying window size events (number of events) to determine the access decision. However, a large number of events will require more time to authenticate the user.

However, the size should be as small as possible to increase the detection speed of unauthorized users. Thus, the goal is to determine the optimal number of events that lead to better results. The access decision of the new request is made at the decision unit. This decision is made based on the classified events of the last accessed apps to be accessed immediately before the access request is sent from the user.

This decision (dec_i) is made based on the last three accessed events, ($app_{i-2}, app_{i-1} and app_i$). Consequently, when the last three classified events are identified for a specific user ($user_i$), the next access request will be accepted.

If the last aessed events are identified to the current user, the nt request will be accepted, otherwise it will be denied, and the user will be requested to undergo a second-factor authentication as proof of identity. For

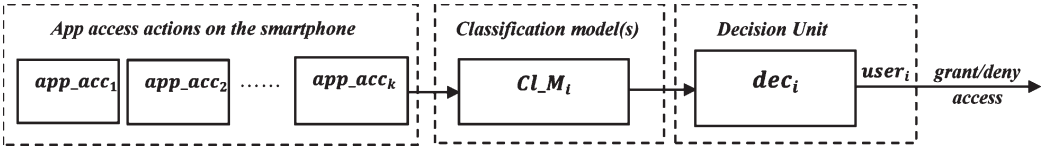

Fig. 3. User authentication procedure.

user authentication, the presented scheme monitors the app access events on smartphone devices to detect abnormalities in user access patterns. Hence, when detecting a deviation in access events, the scheme rejects the next access request and requires a second authentication factor. In the case of providing a second authentication factor from the user, the user pattern template will be updated with this event.

4. Evaluation and results

In order to evaluate the proposed scheme, the dataset UbiqLog4UCI [40], which was collected from 35 users during their regular daily access to apps on their mobile phones, is utilized. The following subsections provide a dataset analysis and preprocessing, preliminary statistical analysis of the extracted features, the evaluation metrics, and the evaluation of the presented method. The scheme is continuously evaluated, based on the FPR and the FNR, for a sequence of three months.

4.1. Dataset analysis

The dataset contains, 66,704 instances (events) for the selected users, as well as approximately 1,415 unique apps. The average interaction time during weekdays and weekends are different. Hence, there is a difference in interaction with apps between weekdays and weekends. In addition, 44% of weekend accessed apps have more access time than during weekdays.

By analyzing the dataset, the average access per user, during both weekdays and weekends, can be determined in comparison with other users. From the analysis, it is clear that the first ranked application (*com.android.nfc*) is the most used, with the second and third apps (*com.sec.android.provider.logsprovider*, and *com.android.settings*) as the second ranked. In addition, the rest of the apps (*com.samsung.android.providers.context*, *com.sec.android.app.twdvfs*, *com.broadcom.bt.app.system*, *com.android.mms*, *com.sec.android.app.launcher*, and *com.sec.pcw.device*) that are ranked from fourth to tenth are in close usage, which means that the usage of these apps is regular among users. As seen from Table 1, the observations for each class represent a minority of the dataset. In addition, all classes are variant, thus considering such variance is necessary in order to avoid ignoring the minority classes over the majority classes.

Table 1
The class representation observations in the dataset

User	Representation observations
1	6217
2	8054
3	8112
4	8771
5	335
6	611
7	22705
8	11789
9	7799
10	1391
11	4310
12	8859
13	6077
14	1824
15	7221
16	18300
17	7309
18	2519
19	17795
20	752

4.2. Preliminary statistical analysis of the extracted features

This subsection discusses the feasibility of utilizing the generated features for differentiating users as well as the feasibility of utilizing these features in user authentication. Once the mentioned features are extracted, a user's behavior pattern can be learned, and a template of this behavior can be built, then utilized for continuous user authentication. Selecting the most effective features is an important process because it will strengthen the pattern template of the user and subsequently affect the performance of the classification process. The extracted features, in

Table 2
Statistical analysis of the extracted features

	Time Duration (Seconds)		Day of year		Weekday		Hour of the day		Day interval		Week interval	
User	*mean*	*std*	*mean*	*std*	*mean*	*std*	*mean*	*Std*	*mean*	*std*	*mean*	*std*
4	25.84	10.28	59.34	75.09	3.02	2.01	15.11	5.79	4.27	1.37	1.99	0.76
5	51.78	10.85	90.00	11.37	2.75	1.93	11.20	7.73	3.26	1.90	2.09	0.72
6	37.16	10.94	83.72	10.00	3.51	2.03	14.84	5.16	4.07	1.34	2.22	0.85
7	27.72	10.51	54.99	80.20	2.98	2.11	15.38	6.22	4.38	1.46	1.87	0.72
8	22.34	10.32	56.16	79.10	2.08	2.08	14.92	6.28	4.32	1.41	1.88	0.68
13	42.45	12.39	49.69	82.53	2.79	1.85	11.10	7.77	3.41	1.94	2.14	0.73
14	25.38	10.64	64.98	60.97	2.61	2.02	14.56	6.31	4.13	1.54	2.08	0.71
15	25.89	10.41	67.15	70.61	3.46	1.91	14.73	6.12	4.11	1.49	1.94	0.8
18	27.55	10.47	46.88	85.45	3.13	2.10	14.38	6.09	4.14	1.42	1.95	0.79
20	23.78	11.12	10.00	89.18	3.30	1.89	12.88	6.95	4.02	1.57	1.95	0.78

addition to the day of the year and day of the week, include hour of the day and day and week intervals.

Hence, in order to test the similarities between users and to examine the effect of the generated features on discriminating between users, the comparison regarding the standard deviation (*std*) and the *mean* are considered, as shown in Table 2. The analysis is shown according to whether it is within the same feature or with other features. In this table, where we choose the most similar users, there is a significant difference between the rest of the users. For the same features, the time duration in accessing apps, as presented in Table 2, shows that the mean is different for users (5, 6, 13, 18 and 20). However, it is similar for users (14 and 15).

In addition, the *std* for users (4–8) and (14–18) is similar. Since they share an almost identical *std*, it is difficult to differentiate between these users using this feature. Thus, in the next feature, the day of the year, the mean and the *std* are mostly different. However, there are similarities between some of the users within the features of weekday and hour of the day. It is observed that the similarities become less when considering the features of day intervals and week intervals, which eliminate the similarities between these users and, in turn, will enhance the performance of the classification.

4.3. Evaluation metrics

My evaluation metrics can be utilized, including: true positive rate (*TPR*); false positive rate (*FPR*); true negative rate (*TNR*); and false negative rate (*FNR*). Other metrics that can also be used include positive predictive value (*PPV*) and negative predictive value (*NPV*):

$$TPR = \frac{TP}{TP + FN} \tag{3}$$

$$TNR = \frac{TN}{TN + FP} \tag{4}$$

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

$$FNR = \frac{FN}{FN + TP} \tag{6}$$

In contrast, most classifiers are more sensitive in detecting majority classes than minority examples, which are the most important in our case. Moreover, in this case, the classifier may be biased. Given a dataset with imbalanced class distribution, as shown in Table 1, the classification performance on a small class usually leads to unsatisfactory results. The classification accuracy metric may not reflect the real performance of the proposed scheme; hence the literature recommends other advanced metrics, such as recall and precision, to evaluate the accuracy of the classification algorithms [41, 42]. The precision (P) is calculated as shown in Equation 7.

$$P = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{7}$$

The measures, the recall (*R*), the sensitivity or the true positive rate (*TPR*), and *P*, are then used in calculating the F-measure to give a single accuracy measurement of a class, ensuring that both measures, the recall and the precision, are reasonably high:

$$\text{F-measure} = \frac{2\mathrm{RP}}{\mathrm{R} + \mathrm{P}} \tag{8}$$

The weighted F-measure ($\bar{F}$) can be calculated from the ratio of the sum of the product of the weight (w_i) times the *F-measure* (F_i), then divided by the sum of the weight (w_i), as follows:

$$\bar{F} = \frac{\sum_{i=1}^{n} w_i.F_i}{\sum_{i=1}^{n} w_i} = \frac{w_1F_1 + w_2F_2 + \ldots w_nF_n}{w_1 + w_2 + \ldots + w_n} \tag{9}$$

4.4. Evaluation results

After building the normal user behavior pattern, the second step is to test the presented scheme in differentiating between users utilizing one-vs-all is applied. Hence, we will apply two methodologies: one-vs-all classification, testing each user against the rest as illegitimate, in addition to keeping some part of the dataset out of the training stage and testing the scheme against this part of the dataset. Hence, in the evaluation process of the presented method, we apply three evaluation criteria: testing the scheme against the test set; testing the scheme against unseen data from the dataset; and then continuously testing the access decision for user authentication based on the presented method by the decision unit.

This decision is made based on the last three accessed events, meaning that when the last three classified events are identified for a specific user, the next access request will be accepted. The equations (5) and (6) are used for the evaluation.

4.4.1. Evaluation based on the FPR and FNR

For the first evaluation step, the scheme is tested based on th set during the training, and the result is shown in Fig. 4. From this figure, it is clear that there is variance of FPR among users, which is due to some similarities between some users as can be seen from the analysis in Table 2. In addition, the FNR is close to zero for all users, as shown in Fig. 5. However, when testing the scheme on the unseen data, the FNR increases while the FPR still registers almost no change. In spite of the fact that we need to reduce both the FPR and the FNR, the effect of the FNR will be on the usability of the scheme regarding the request for second-factor authentication. The results, as in Fig. 5, show that the scheme still produces low FNR when applied to unseen users' data. Also, the increase in FNR values, for example, for users (5, 6, 9, 13, 14, 18, and 20) is because there is close usage between these users. As an example, for users (14 and 15), the *mean* and *std* are close regarding the feature time duration, hour of the day and day interval. However, the difference can be recognized for the day of the week and year interval features.

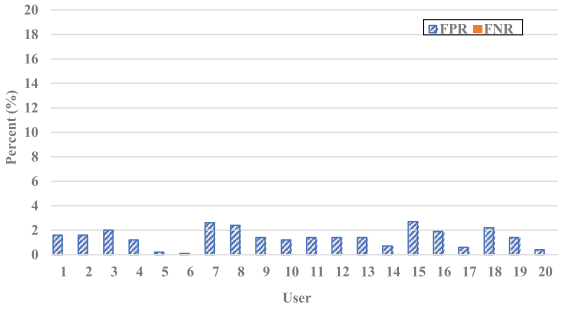

Fig. 4. Scheme performance on test set.

4.4.2. Evaluation based on simulated access requests

In this evaluation, access requests are simulated on unseen data samples. There is a variance of the number of events accessed by users, and the test is achieved based on equations (1) and (2). However, the total number of events of users is different, and the result is shown in Fig. 6. From this figure, we can see that the minimum percent of average access decisions made is 91.87 % for user 15 while the maximum percent of average access decisions made is 99.49 for user 6. The low 91.87 % for user 15 is because of the similarity with user 14. The same result can be seen with users (7 and 8), where the *mean* and *std* are almost identical regarding the feature time duration, hour of the day, day interval, and week interval. However, the difference can be recognized only with the day of year feature.

4.5. Discussion

The results demonstrate that users can be continuously authenticated based on their app usage time during the selected time period with considerably high accuracy. We were also able to build a behavior pattern template with low FPRs and FNRs for user authentication during a period of three months, the time during which the dataset was collected. Although there are some changes in app access time, these changes do not significantly affect accuracy due to the inclusion of the extracted features. Although there is still some FNR that minimally affects the usability of the presented method, the effect is only on usability, which increases the request of second-factor authentication. In this paper, we consider only the authentication aspect and assume that all users

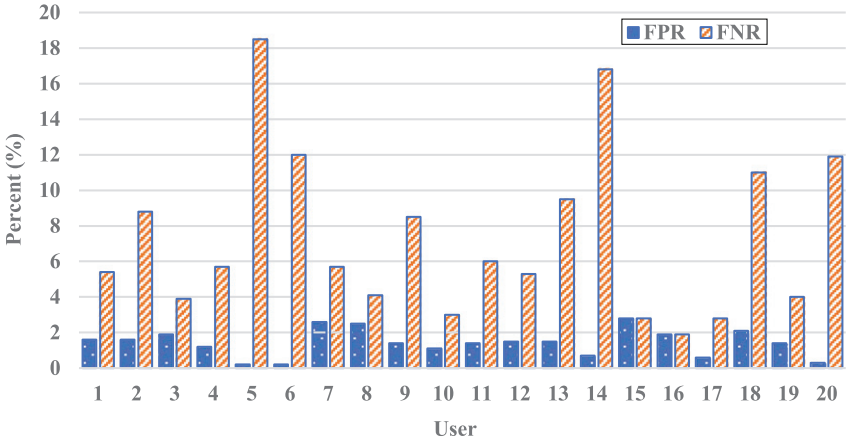

Fig. 5. Scheme performance on unseen data.

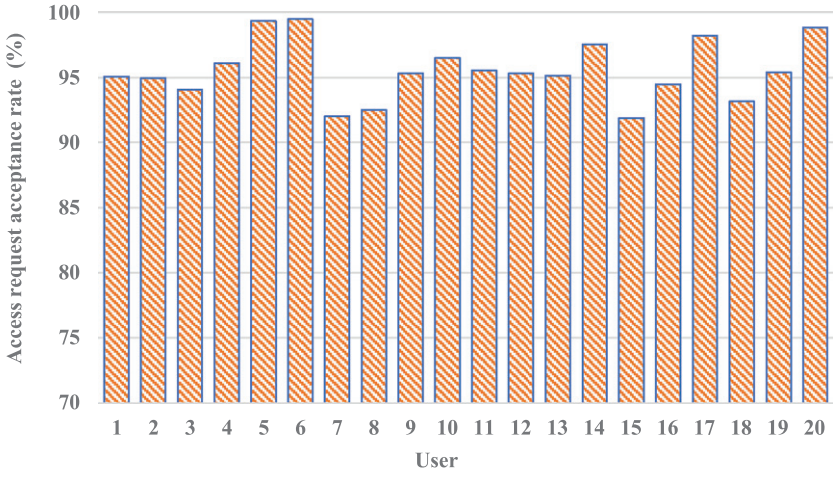

Fig. 6. Scheme performance on simulated access requests.

are registered and known. However, identification is also an important issue that has to be considered in order to detect unauthorized access from known users (insiders). Although it is challenging to build a behavior pattern template for unknown users, this issue will be the focus of future work. The study in [16] considered traffic generated app access pattern and concluded that users can be authenticated based on this pattern. However, traffic generated information is not considered in this work due to the nature of the utilized dataset. In a future study, we believe the availability of this feature will considerably improve the accuracy of the authentication.

5. Conclusion and future work

The increasing number of smartphone apps, as well as the opportunity to constantly install additional apps on mobile devices, will provide the possibility to discriminate and continuously authenticate users with proper accuracy. In this paper, we provided an extension of a previous paper and presented a machine learning user authentication scheme based on app interactions. The scheme evaluation is performed on a dataset based on both a testing set and unseen data, and the results demonstrate the capacity of the presented method to continuously authenticate users with a considerably low FPR and FNR. The resulting FPR and FNR rates, especially when applying unseen data, can be eliminated by extracting more features, such as the app access order within the same apps as well as within other apps., in order to reduce the similarity between users. For future work, we plan to evaluate our scheme on other datasets with a larger number of users. Additionally, we plan to extract more features, such as the

app access order for in order to reduce the similarity among users.

References

[1] H. Cao and M. Lin, Mining Smartphone Data for app Usage Prediction and Recommendations: A Survey, *Pervasive and Mobile Computing* **37** (2017), 1–22.

[2] E. Fernandes, J. Jung and A. Prakash, Security Analysis of Emerging Smart Home Applications, *Proceedings – 2016 IEEE Symposium on Security and Privacy, SP 2016* (2016), 636–654.

[3] C. Meffert, D. Clark, I. Baggili and F. Breitinger, Forensic State Acquisition From Internet of Things (FSAIoT): A General Framework and Practical Approach for IoT Forensics Through IoT Device State Acquisition, in *12th International Conference on Availability, Reliability and Security*, 2017, pp. 1–11.

[4] Y. Ashibani and Q. H. Mahmoud, A Machine Learning-Based User Authentication Model Using Mobile App Data, in *International Conference on Intelligent and Fuzzy Systems (INFUS)*, 2019, pp. 408–415, (Best Paper Award).

[5] Z.M. Zhang, Lide; Tiwana, Birjodh; Qian, Zhiyun;Wang, Zhaoguang;Dick, Robert P.; Mao and L. Yang, Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones, in *Proceedings of the eighth IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis.*, 2010, pp. 105–114.

[6] F. Li, N. Clarke, M. Papadaki and P. Dowland, Behaviour Profiling for Transparent Authentication for Mobile Devices, *European Conference on Cyber Warfare and Security. Academic Conferences International Limited*, pp. 307–315, 2011.

[7] E. Shi, Y. Niu, M. Jakobsson and R. Chow, Implicit Authentication Through Learning User Behavior, *Springer, Berlin, Heidelberg.*, pp. 99–113, 2011.

[8] D. Damopoulos, S. A. Menesidou, G. Kambourakis, M. Papadaki, N. Clarke and S. Gritzalis, Evaluation of Anomaly-Based IDS for Mobile Devices Using Machine Learning Classifiers, *Security and Communication Networks* **5**(1) (2012), 3–14.

[9] S. Alotaibi, A. Alruban, S. Furnell and N. Clarke, A Novel Behaviour Profiling Approach to Continuous Authentication for Mobile Applications, in *The 5th International Conference on Information Systems Security and Privacy*, 2019, no. February, pp. 1–6.

[10] L. Fridman, S. Weber, R. Greenstadt and M. Kam, Active Authentication on Mobile Devices via Stylometry, Application Usage, Web Browsing, and GPS Location, *IEEE Systems Journal* **11**(2) (2017), 513–521.

[11] T.J. Neal and D.L. Woodard, Using Associative Classification to Authenticate Mobile Device Users, in *IEEE International Joint Conference on Biometrics, IJCB 2017*, 2017, pp. 71–79.

[12] A. Kalamandeen, A. Scannell, E. De Lara, A. Sheth and A. Lamarca, Ensemble: Cooperative Proximity-Based Authentication, pp. 331–343, 2010.

[13] U. Mahbub, J. Komulainen, D. Ferreira and R. Chellappa, Continuous Authentication of Smartphones Based on Application Usage, *IEEE Transactions on Biometrics, Behavior, and Identity Science* **1**(3) (2019), 165–180.

[14] F. Li, N. Clarke, M. Papadaki and P. Dowland, Active Authentication for Mobile Devices Utilising Behaviour Profiling, *International Journal of Information Security* **13**(3) (2014), 229–244.

[15] Y. Ashibani and Q.H. Mahmoud, A Behavior Profiling Model for User Authentication in IoT Networks based on App Usage Patterns, in *44th IEEE Annual Conference of the Industrial Electronics Society (IECON)*, 2018, 2841–2846.

[16] Y. Ashibani and Q.H. Mahmoud, A User Authentication Model for IoT Networks Based on App Traffic Patterns, in *9th IEEE Annual I Information Technology; Electronics and Mobile Communication Conference (IEMCON)*, 2018, pp. 632–638.

[17] J. Liu, L. Zhong, J. Wickramasuriya and V. Vasudevan, uWave: Accelerometer-Based Personalized Gesture Recognition and its Applications, *Pervasive and Mobile Computing* **5**(6) (2009), 657–675.

[18] L. Li, X. Zhao and G. Xue, Unobservable Re-authentication for Smartphones, *NDSS – Network and Distributed System Security Symposium*, pp. 1–16, 2013.

[19] N. Sae-Bae, N. Memon, K. Isbister and K. Ahmed, Multitouch Gesture-Based Authentication, *IEEE Transactions on Information Forensics and Security* **9**(4) (2014), 568–582.

[20] H. M. Thang, V. Q. Viet, N. Dinh Thuc and D. Choi, Gait Identification Using Accelerometer on Mobile Phone, *International Conference on Control, Automation and Information Sciences (ICCAIS)*, pp. 344–348, 2012.

[21] Sangil Choi, Ik-Hyun Youn, R. LeMay, S. Burns and J.-H. Youn, Biometric Gait Recognition Based on Wireless Acceleration Sensor Using K-Nearest Neighbor Classification, *International Conference on Computing, Networking and Communications (ICNC)*, pp. 1091–1095, 2014.

[22] A. Kale, A.N. Rajagopalan, N. Cuntoor and V. Krüger, Gait-Based Recognition of Humans Using Continuous HMMs, *Proceedings – 5th IEEE International Conference on Automatic Face Gesture Recognition, FGR*, pp. 336–341, 2002.

[23] D. Gafurov, K. Helkala and T. Søndrol, Biometric Gait Authentication Using Accelerometer Sensor, *Journal of Computers* **1**(7) (2006), 51–59.

[24] C. Nickel, T. Wirtl and C. Busch, Authentication of Smartphone Users Based on the Way They Walk Using K-NN Algorithm, *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), Eighth International Conference, IEEE,* pp. 16–20, 2012.

[25] J. Hall, M. Barbeau and E. Kranakis, Anomaly-Based Intrusion Detection Using Mobility Profiles of Public Transportation Users, *International Conference on Wireless And Mobile Computing, Networking And Communications (WiMob'2005), IEEE.* **2** (2005), 17–24.

[26] S. Subudhi and S. Panigrahi, Quarter-Sphere Support Vector Machine for Fraud Detection in Mobile Telecommunication Networks, *Procedia Computer Science* **48** (2015), 353–359.

[27] B. Al-Bayati, N. Clarke and P. Dowland, Adaptive Behavioral Profiling for Identity Verification in Cloud Computing: A Model and Preliminary Analysis, *GSTF Journal on Computing* **5**(1) (2016), 21–28.

[28] A. Aupy and N. Clarke, User Authentication by Service Utilisation Profiling, *Advances in Network and Communications Engineering* **2**(18) (2005).

[29] S. Yazji, X. Chen, R.P. Dick and P. Scheuermann, Implicit User Re-Authentication for Mobile Devices, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **5585**(LNCS) (2009), 325–339.

[30] M. Ben Salem and S.J. Stolfo, Modeling User Search Behavior for Masquerade Detection, *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **6961**(LNCS) (2011), 181–200.

[31] Y. Ashibani, D. Kauling and Q.H. Mahmoud, Design and Implementation of a Contextual-Based Continuous Authentication Framework for Smart Homes, *Applied System Innovation* **2**(1) (2019), 1–20.

[32] L. Abdi and S. Hashemi, To Combat Multi-Class Imbalanced Problems by Means of Over-Sampling Techniques, *IEEE Transactions on Knowledge and Data Engineering* **28**(1) (2016), 238–251.

[33] K.S. Kim, H.H. Choi, C.S. Moon and C.W. Mun, Comparison of K-Nearest Neighbor, Quadratic Discriminant and Linear Discriminant Analysis in Classification of Electromyogram Signals Based on the Wrist-Motion Directions, *Current Applied Physics* **11**(3) (2011), 740–745.

[34] Tiago A. Almeida and A. Yamakami, Compression-Based Spam Filter, *Security and Communication Networks* **9**(4) (2016), 1327–335.

[35] T. Windeatt, Accuracy/Diversity and Ensemble MLP Classifier Design, *IEEE Transactions on Neural Networks* **17**(5) (2006), 1194–1211.

[36] M. Li, et al., Coupled K-Nearest Centroid Classification for Non-iid Data, in *Transactions on Computational Collective Intelligence XV*, Springer, Berlin, Heidelberg., 2014, pp. 89–100.

[37] M.F. Amasyali and O.K. Ersoy, Classifier Ensembles with the Extended Space Forest, *IEEE Transactions on Knowledge and Data Engineering* **26**(3) (2014), 549–562.

[38] Y. Qi, Random Forest for Bioinformatics, *Ensemble Machine Learning: Methods and ApplicatiOns*, pp. 307–323, 2012.

[39] J. Xia, P. Ghamisi, N. Yokoya and A. Iwasaki, Random Forest Ensembles and Extended Multiextinction Profiles for Hyperspectral Image Classification, *IEEE Transactions on Geoscience and Remote Sensing* **56**(1) (2017), 202–216.

[40] R. Rawassizadeh, E. Momeni, C. Dobbins and P. Mirzababaei, Lesson Learned from Collecting Quantified Self Information via Mobile and Wearable Devices, *Journal of Sensor and Actuator Networks* **4**(4) (2015), 315–335.

[41] V. García, J.S. Sánchez and R.A. Mollineda, Knowledge-Based Systems On the Effectiveness of Preprocessing Methods When Dealing with Different Levels of Class Imbalance, *Knowledge-Based Systems, Elsevier* **25** (2012), 13–21.

[42] Y. Sun, A.K.C. Wong and M.S. Kamel, Classification of Imbalanced Data: A Review, *International Journal of Pattern Recognition and Artificial Intelligence* **23**(04) (2009), 687–719.